CLAIMS:

Having thus described our invention, what we claim as new, and desire to secure by Letters Patent is:

1. A computing system for performing high speed data page operations having a processor device for generating real addresses associated with memory locations of a real memory system for reading and writing of data thereto, said system comprising:

a plurality of memory blocks in said real memory system for storing said data,

a physical memory storage for storing the pages of data comprising one or more real memory blocks, each said real memory block partitioned into one or more sectors, each comprising contiguous bytes of said physical memory;

a translation table structure in said physical memory storage having entries for associating a real address with sectors of said physical memory, each translation table entry including one or more pointers for pointing to a corresponding sector in its associated real memory block, said table accessed for storing data in one or more allocated sectors for memory read and write operations initiated by said processor;

a control device for directly manipulating entries in said translation table structure for performing block swap, block move and block clear page operations for all blocks in a page without actually accessing physical memory data contents.

2. The computing system as claimed in Claim 1, wherein a page operation includes moving blocks associated from a source page to a destination page, said control device directly moving sector pointers from entries of each block associated with a source page in said translation table to entries in said translation table of blocks associated with a destination page.

3. The computing system as claimed in Claim 2, further comprising a first memory mapped register for specifying addresses associated with said source and destination

pages involved in a page operation, said addresses specified by said first register accessible by a system processor device (CPU).

4. The computing system as claimed in Claim 3, further comprising a second memory mapped register accessible by said CPU for commanding specific page operations and reporting of their status.

5. The computing system as claimed in Claim 4, wherein said second memory mapped register accessible by said CPU includes a data field for receiving a function code by said CPU for commanding a particular page operation.

6. The computing system as claimed in Claim 4, wherein said second memory mapped register accessible by said CPU includes a status bit field for indicating a status of said page operation, said status including one of: a busy status and a completion status.

7. The computing system as claimed in Claim 3, wherein said control device further comprises mechanism for suspending memory accesses to source and destination pages while a page operation is performed.

8. The computing system as claimed in Claim 5, wherein said control device further comprises mechanism for dynamically allocating sectors in a memory block and calculating an index into an allocated sector for performing data read and data write operations thereto, said control device implementing a free list structure having address pointers for pointing to unallocated sectors to be allocated.

9. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a page operation requiring pointers for memory sectors of a page referenced by a source page to be returned to said free list structure.

10. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a page operation requiring pointers for memory sectors of a source page to be returned to the free list structure.

11. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a page operation that requires sectors of the physical memory of a destination page to be moved to sectors associated with physical memory of a source page.

12. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a page operation that requires addition or deletion of physical memory sectors of a destination page such that its physical size is equivalent to a source destination page.

13. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a copy page operation wherein pointers to sectors of physical memory referenced by a source page are copied to pointers for sectors of physical memory referenced by a destination page.

14. The computing system as claimed in Claim 8, wherein a function command code specified in said second register indicates a swap page operation wherein pointers of sectors of physical memories referenced by source page and destination page are exchanged.

15. The computing system as claimed in Claim 8, further comprising a caching hierarchy of one or more cache devices associated with said CPU for caching memory data contents, wherein a function command code specified in said second register indicates a page operation requiring the memory addresses associated with a source or destination page to be invalidated in the memory caching hierarchy, wherein cached portions of a

source page and destination page are invalidated in said cache devices as part of a page operation.

16. The computing system as claimed in Claim 15, wherein a function command code specified in said second register indicates a page operation requiring the memory addresses associated with a source or destination page to be flushed from the memory caching hierarchy, wherein cached portions of said source or destination pages are flushed from said cache devices as part of a page operation.

17. The computing system as claimed in Claim 16, wherein said second memory mapped register accessible by said CPU includes a destination coherent bit field for indicating whether coherency or partial coherency is to be maintained during a page operation, said partial coherency enabling cache flush or invalidate actions to be omitted for a destination page.

18. The computing system as claimed in Claim 5, further comprising a translation table buffer device accessible by said CPU for storing translation table entries and accessible by said CPU using normal memory access instructions, wherein a function command code specified in said second memory mapped register initiates copying of contents of a translation table entry for a source page to said buffer device.

19. A method for performing high speed data page operations in a computer system comprising a real system memory and including a processor device for generating real addresses associated with memory locations of said real memory system for reading and writing data thereto, said method comprising:

a) providing a physical memory storage for storing pages of data comprising one or more real memory blocks, each said real memory block comprising one or more sectors, each comprising contiguous bytes of said physical memory;

b) maintaining a translation table structure in said physical memory storage having entries for associating a real address with sectors of said physical memory, each translation table entry including one or more pointers for pointing to a corresponding sector in its associated real memory block, said table accessed for storing data in one or more allocated sectors for memory read and write operations initiated by said processor;

c) directly manipulating entries in said translation table structure for performing a block swap, block move and block clear page operation for all blocks in a page without actually accessing physical memory data contents.

20. The method as claimed in Claim 19, wherein a page operation includes a moving blocks associated from a source page to a destination page, said manipulating step c) including directly moving sector pointers from entries of each block associated with a source page in said translation table to entries in said translation table of blocks associated with a destination page.

21. The method as claimed in Claim 20, wherein said computer system includes a first memory mapped register accessible by a CPU for specifying addresses associated with said source and destination pages involved in a page operation, said method including the step of specifying addresses for said CPU access.

22. The method as claimed in Claim 21, wherein said computer system includes a second memory mapped register accessible by said CPU for commanding specific page operations, said method further including the step of generating for receipt by said second memory mapped register a function code for commanding a particular page operation.

23. The method as claimed in Claim 22, further including the step of indicating a status of a page operation in a status bit field provided in said second memory mapped register for access by said CPU, said status bit field indicating one of: a busy status and a completion status.

24. The method as claimed in Claim 20, further including the step of suspending memory accesses to source and destination pages while a page operation is performed.

25. The method as claimed in Claim 22, further including the steps of:

dynamically allocating sectors in a memory block and calculating an index into an allocated sector for performing data read and data write operations thereto; and,

implementing a free list structure having address pointers for pointing to unallocated sectors to be allocated.

26. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a page operation requiring pointers for memory sectors of a page referenced by a source page to be returned to said free list structure.

27. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a page operation requiring pointers for memory sectors of a source page to be returned to the free list structure.

28. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a page operation that requires sectors of the physical memory of a destination page to be moved to sectors associated with physical memory of a source page.

29. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a page operation that requires addition or deletion of physical memory sectors of a destination page such that its physical size is equivalent to a source destination page.

30. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a copy page operation requiring pointers to sectors of physical memory referenced by a source page to be copied to pointers for sectors of physical memory referenced by a destination page.

31. The method as claimed in Claim 25, further including the step of: specifying a function command code in said second register for indicating a swap page operation requiring an exchange of pointers to sectors of physical memories referenced by source page and destination page.

32. The method as claimed in Claim 25, wherein said computer system further comprises a caching hierarchy of one or more cache devices associated with said CPU for caching memory data contents, said method including the step of: specifying a function command code in said second register for indicating a page operation requiring the memory addresses associated with a source or destination page to be invalidated in the memory caching hierarchy, wherein cached portions of a source page and destination page are invalidated in said cache devices as part of a page operation.

33. The method as claimed in Claim 32, further including the step of: specifying a function command code in said second register for indicating a page operation requiring the memory addresses associated with a source or destination page to be flushed from the memory caching hierarchy, wherein cached portions of said source or destination pages are flushed from said cache devices as part of a page operation.

34. The method as claimed in Claim 33, wherein said second memory mapped register accessible by said CPU includes a destination coherent bit field for indicating whether coherency or partial coherency is to be maintained during a page operation, said method further including the step of: indicating a page operation requiring said partial coherency, whereby cache flush or invalidate actions are omitted for a destination page.

35. The method as claimed in Claim 33, further including the step of: specifying a function command code in said second register for initiating copying of contents of a translation table entry for a source page to a translation table buffer device accessible by said CPU for storing a translation table entry.